



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/008,952	12/06/2001	Ashley K. Wise	RA5417 (USYS030.PA)	3730

27516 7590 12/27/2004

UNISYS CORPORATION  
MS 4773  
PO BOX 64942  
ST. PAUL, MN 55164-0942

EXAMINER
----------

MITCHELL, JASON D

ART UNIT	PAPER NUMBER
----------	--------------

2124

DATE MAILED: 12/27/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

<b>Office Action Summary</b>	<b>Application N</b> 10/008,952	<b>Applicant(s)</b> WISE, ASHLEY K.	
	<b>Examiner</b> Jason Mitchell	<b>Art Unit</b> 2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

#### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

#### Status

- 1) ☒ Responsive to communication(s) filed on 06 December 2001.
- 2a) ☐ This action is **FINAL**.                      2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

#### Disposition of Claims

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

#### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 06 December 2001 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

#### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All    b) ☐ Some \*    c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

#### Attachment(s)

- |  |   |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892)  | 4) <input type="checkbox"/> Interview Summary (PTO-413)<br>Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948)                                   | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152)             |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)<br>Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____  |

### DETAILED ACTION

1. This action is in response to an application filed on 12/06/2001.
2. Claims 1-20 are pending in this case.

### ***Claim Rejections - 35 USC § 102***

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. **Claims 1-4, 11-12, 14 and 16-20 are rejected under 35 U.S.C. 102(b) as being anticipated by White.**

**Regarding Claims 1 and 18:** White discloses a computer-implemented method for processing numerical values in a computer program executable on a computer system, comprising: encapsulating in a large-integer datatype, large-integer data and associated operators (pg. 174, col. 1, par. 3 'indefinitely large integers—have been a part of Lisp for a long time'), wherein the large-integer data has runtime expandable precision (pg. 177, col. 2, par. 2 'allocated in units of at least one 32-bit word') and maximum precision is limited only by system memory availability (pg. 174, col. 1, par. 3 'indefinitely large integers'); and overloading language-provided arithmetic, logical, and type conversion operators with the large-integer operators that operate on large-integer variables in combination with other datatypes, and programmed usage of a variable of the large-integer datatype is equivalent to and interoperable with a variable of a system-defined

Art Unit: 2124

integral datatype (pg. 174, col. 1, par. 3 - col. 2, par. 1 'a smooth, user invisible transition between ... fixnums—and those of larger size').

**Regarding Claim 2:** The rejection of claim 1 is incorporated; further White discloses that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

"Common Lisp The Language, 2<sup>nd</sup> Edition" by Guy Steel et al. (Common Lisp) teaches converting a character string into large-integer data in response to a constant definition statement (sec. 5.1.1, par 1 'all numbers ... are self-evaluation forms. When such an object is evaluated, that object ... is returned as the value of the form').

Therefore, through his use of COMMON LISP, White inherently discloses converting a character string into large-integer data in response to a constant definition statement.

**Regarding Claim 3:** The rejection of claim 2 is incorporated; further White discloses that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

Common Lisp teaches converting large-integer data to and from a character string for input (sec. 22.1.1 par. 16 'After the entire token is read in, it will be interpreted either as ... or number'), output (sec. 22.1.6, par. 2-3 'How an expression is printed depends on its data type'), and serialization (sec. 22.1.1 par. 16 'it begins an extended token. After the entire token is read in').

Therefore, through his use of COMMON LISP, White inherently discloses converting large-integer data to and from a character string for input, output, and serialization.

**Regarding Claim 4:** The rejection of claim 1 is incorporated; further, White discloses that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

Common Lisp teaches converting input data from language-provided input functions to large-integer data (sec. 22.1.1 par. 15 'After the entire token is read in, it will be interpreted either as ... or number'); and converting large-integer data to a format compatible with language-provided output functions (sec. 22.1.6, par. 2-3 'How an expression is printed depends on its data type').

Therefore, through his use of COMMON LISP, White inherently discloses converting input data from language-provided input functions to large-integer data and converting large-integer data to a format compatible with language-provided output functions.

**Regarding Claim 11:** The rejection of claim 1 is incorporated; further White discloses the algorithm used for division is defined in "The Art of Computer Programming, Vol. II" by Knuth (Knuth) (pg. 177, col. 1, par. 2 'The particular algorithms used are ...described in section 4.3.1 of [Knuth 1981])

Knuth teaches identifying a set of most-significant bits of the dividend and a set of least-significant bits of the dividend (pg. 257, Algorithm D ' $v=(v_1v_2...v_n)_b$ '); recursively performing a large-integer divide operation using the set of most-significant bits as the input dividend (pg. 257, Algorithm D Step D2 'a division of  $(u_ju_{j+1}...u_{j+n})_b$ ', and returning a quotient and a remainder (Algorithm D step D4 'replace  $(u_ju_{j+1}...u_{j+n})_b$  by  $(u_ju_{j+1}...u_{j+n})_b$  minus  $q$  times  $(v_1v_2...v_n)_b$ '); finding a lower-part dividend as a function of the remainder and the set of least-significant bits (Algorithm D step D7 'increase  $j$  by one'); recursively

Art Unit: 2124

performing a large-integer divide operation using the lower-part dividend (Algorithm D steps D7 'go back to D3'); and concurrently solving for the quotient and the remainder (Algorithm D step D4 'minus q times  $(v_1v_2...v_n)_b$ ').

Therefore, through the use of the algorithm taught by Knuth, White implicitly discloses the limitations recited.

**Regarding Claim 12:** The rejection of claim 11 is incorporated; further White discloses identifying an optimal set of most-significant bits of the dividend and a set of least-significant bits of the dividend as a function of a number of bits that represent the dividend and a number of bits that represent the divisor (pg. 178, col. 1, par. 3 'the size of a bigit ... will vary from ... algorithm to algorithm').

**Regarding Claim 14:** The rejection of claim 1 is incorporated; further White discloses emulating fixed-bit arithmetic on variables of the large-integer data type (pg. 174, col. 2, par. 4 'a set of new primitive arithmetic operations that focus on ... bignums').

**Regarding Claims 17 and 20:** The rejections of claims 1 and 18 are incorporated; further White does not explicitly disclose a large-rational datatype, but does disclose that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

Common Lisp teaches a rational datatype (sec. 2.1.2, par. 1 'Integers and ratios collectively constitute the type rational') where a ratio is the mathematical ratio of two integers (sec. 2.1.2, par. 1), and integers encompass bignums (sec. 2.1.1, par. 2 'an integer that is not a fixnum is called a bignum'), thereby teaching the limitations recited in the instant claim as noted in the rejections of claims 1 and 18.

Therefore, through his use of COMMON LISP, White inherently discloses a large-rational datatype.

***Claim Rejections - 35 USC § 103***

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

6. **Claims 5-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over 'Reconfigurable, Retargetable Bignums' by White (White) in view of US 5,446,901 to Owicki et al. (Owicki).**

**Regarding Claim 5:** The rejection of claim 1 is incorporated; further White discloses for each large-integer variable having a value other than zero, storing a numerical value in at least one storage node allocated to the variable (pg. 176, col. 2, par. 3 'resemble simple bit vectors'). However White does not explicitly disclose establishing a plurality of storage nodes for allocation to large-integer data, although he does indicate the existence of language provided primitives that do (pg. 176, col. 2, par. 3 'primitives ... to allocate memory for one of a given size').

Owicki discloses establishing a plurality of storage nodes for allocation to large-integer data (col. 1, lines 17-20 'pool of memory') in an analogous art for the purpose of maintaining a pool of available memory (col. 1, lines 17-20 'memory available for allocation').

Art Unit: 2124

It would have been obvious to a person of ordinary skill in the art at the time of the invention to implement memory allocation/de-allocation in White's invention using the teachings of Owicki (col. 1, lines 17-20), because one of ordinary skill in the art would have been motivated to allocate and de-allocate memory to and from objects being created and destroyed (pg. 177, col. 2, par 2 'Bignums are allocated in units of at least on 32-bit word').

**Regarding Claim 6:** The rejection of claim 5 is incorporated; further, White discloses allocating a selected number of bits for each storage node in response to a program-specified parameter (pg. 178, col. 1, par. 3 'the size of a bigit, ... will vary from implementation to implementation').

**Regarding Claim 7:** The rejection of claim 6 is incorporated; further, White discloses dynamically allocating a number of storage nodes for storage of the numerical value as a function of a size of the numerical value (pg. 177, col. 2, par 2 'Bignums are allocated in units of at least one 32-bit word').

**Regarding Claim 8:** The rejection of claim 7 is incorporated; further, White discloses storing in each node that is allocated to large-integer variable, a subset of bit values that represent a numerical value (pg. 178, col. 1, par. 3 'a 'bigit' is a 'bignum digit' and is thus an integer between 0 and R-1 for some positive radix R').

**Regarding Claim 9:** The rejection of claim 8 is incorporated; further, White does not explicitly address the implementation details of memory allocation/de-allocation except to note that memory is allocated (pg. 177, col. 2, par 2 'Bignums are allocated in units of at least on 32-bit word')



Art Unit: 2124

Owicki teaches maintaining a set of available storage nodes that are not allocated to any object (col. 1, lines 17-20 'the pool of memory'); allocating a storage node from the set of available storage nodes a new object (col. 1, lines 17-20 'available for allocation to new objects'); and returning to the set of available storage nodes a storage node allocated to an unused object (col. 1, lines 17-20 'return memory storage currently allocated to inaccessible objects to the pool'), in an analogous art for the purpose of maintaining a pool of available memory (col. 1, lines 17-20 'memory available for allocation').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to implement memory allocation/de-allocation in White's invention using the teachings of Owiki (col. 1, lines 17-20), because one of ordinary skill in the art would have been motivated to allocate and de-allocate memory to and from objects being created and destroyed (pg. 177, col. 2, par 2 'Bignums are allocated in units of at least on 32-bit word').

**Regarding Claim 10:** The rejection of claim 9 is incorporated; further White discloses that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

Common Lisp inherently discloses overloading language-provided memory allocation and de-allocation operators with large-integer operators that allocate and de-allocate storage nodes. COMMON LISP seeks to make the use of Bignums and Fixnums seamless (sec 2.1.1 par. 'Common Lisp is designed to hide this distinction'), additionally functions such as *let* make no distinction between the two (sec. 7.5, par. 4 "all of the

variables varj are bound to the corresponding values'), thus overloading the memory handling functions.

Therefore, through his use of COMMON LISP, White inherently discloses overloading language-provided memory allocation and de-allocation operators with large-integer operators that allocate and de-allocate storage nodes.

**Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over 'Reconfigurable, Retargetable Bignums' by White (White) in view of "Fast Recursive Division" by Burnikel et al. (Burnikel).**

**Regarding Claim 13:** The rejection of claim 12 is incorporated; further White does not disclose identifying an optimal set of most-significant bits of the dividend and a set of least-significant bits of the dividend as a function one-half a difference between the number of bits that represent the dividend and the number of bits that represent the divisor. However, White does disclose the possibility of using various algorithms (pg. 177, col. 1, par. 2 'we have investigated some more complex algorithms').

Burnikel teaches a division algorithm identifying an optimal set of most-significant bits of the dividend and a set of least-significant bits of the dividend as a function one-half a difference between the number of bits that represent the dividend and the number of bits that represent the divisor (pg. 4, par. 3 'dividing a  $2n$ -digit number by an  $n$ -digit number ... split  $A$  into four parts ... of length  $n/2$  each') in an analogous art for the purpose of improving the processing speed of a divide operation (pg. 1, par. 1 'our algorithm ... yields a speedup of more than 20%')

Art Unit: 2124

It would have been obvious to a person of ordinary skill in the art at the time of the invention to implement the division operation disclosed in White using the algorithm taught in Burnikel (pg. 4, Algorithm 1), because one of ordinary skill in the art would have been motivated to improve performance for a system where most of the division would be done for larger numbers (White pg. 177, col. 1, par. 2 'algorithms that do show a significant improvement in the asymptotic behaviors' and Burnikel pg. 4, par. 2 'Under the assumption that  $n$  is even and large').

**Claims 15-16 and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over 'Reconfigurable, Retargetable Bignums' by White (White) in view of US 5,619,711 to Anderson (Anderson).**

**Regarding Claim 15:** The rejection of claim 1 is incorporated; further White does not disclose transferring data associated with temporary variables of the large-integer datatype by moving pointers to the data.

Anderson teaches transferring data associated with temporary variables of the large-integer datatype by moving pointers to the data, in an analogous art for the purpose of avoiding fragmentation when expanding the data structure of a large-integer (col. 8, lines 59-61 'to avoid fragmentation, a linked list type of allocation may be used for the array')

It would have been obvious to a person of ordinary skill in the art at the time of the invention to utilize the techniques taught in Anderson (col. 8, lines 59-61) when updating large-integer-data as disclosed in White (pg. 174, col. 1, par. 3 'indefinitely large

Art Unit: 2124

integers) because one of ordinary skill in the art would have been motivated to avoid fragmentation as taught in Anderson (col. 8, lines 59-61).

**Regarding Claims 16 and 19:** The rejections of claims 1 and 18 are incorporated; further White does not explicitly disclose a large-floating-point datatype, but does disclose that COMMON LISP is the base language for his invention (pg. 175, col. 1, par. 3 'other commercially available Common Lisp implementations').

Anderson teaches a large-floating-point datatype (col. 4, lines 15-17 'implementing infinite precision binary arithmetic') in an analogous art for the purpose of preserving numerical precision (col. 3, lines 15-17 'for preserving numerical precision').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to use the teachings of Anderson to incorporate a large-floating-point datatype, using the techniques taught in Anderson (col. 4, lines 15-17), in a method similar to that disclosed for Bignums in White (pg. 174, col. 1, par. 3 'indefinitely large integers—have been a part of Lisp for a long time'), because one of ordinary skill in the art would have been motivated to 'provide special operation for floating point math' (Anderson col. 2, lines 5-8), thereby teaching the limitations recited in the instant claim as noted in the rejections of claims 1 and 18.

### **Conclusion**

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. US 4,989,134 to Shaw and "A Portable Extended Precision Arithmetic Package and Library With Fortran Precompiler" by Wyatt et al.


Art Unit: 2124

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jason Mitchell whose telephone number is (571) 272-3728. The examiner can normally be reached on Monday-Thursday and alternate Fridays 7:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571) 272-3719. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jason Mitchell  
12/2/04

  
KAKALI CHAKI  
SUPERVISORY PATENT EXAMINER  
TECHNOLOGY CENTER 2100